AMENDMENTS TO THE CLAIMS

In accordance with the PTO's amendment format, a detailed listing of all claims has been provided. A status identifier is provided for each claim in parentheses following each claim number. Changes to the claims are shown by strikethrough (for deleted text) or underlining (for added text).

In the Claims:

Claims 1-57 were previously pending.

Claims 1, 4, 6, 10, 49, and 54 are currently amended.

No new claims are added.

Claims 2-3 are cancelled.

Claims 1, 4-57 are pending.

1. (Currently Amended) A method for determining a likelihood that a word in a dictionary is being incorrectly represented by a string; comprising:

~~receiving an entered string; and~~

iteratively partitioning the word into multiple segments, each segment consisting of a character or character sequence, wherein each iteration partitions the word into a different number of the multiple segments;

for each iteration of the partitioning, iteratively varying the lengths of the segments while maintaining the number of the segments;

for each iteration of the partitioning, dividing the string into the same number of string segments as the number of word segments and iteratively varying the lengths of the string segments, wherein corresponding word segments and string segments can be of different lengths;

for each iteration of varying the lengths of the word segments and the string segments, computing a probability for each pair, wherein each pair consists of a word segment and a corresponding string segment, and wherein the probability consists of a likelihood that the word segment is being incorrectly represented by the string segment;

for each iteration of varying the lengths, computing a product of the probabilities of the pairs; and

determining ~~how likely a~~ the likelihood that the word ~~was to have been entered as~~ is being incorrectly represented by the string based on one of the products ~~at least one edit operation that converts one of multiple character sequences of arbitrary length in the word to one of multiple character sequences of arbitrary length in the string~~.

2-3. (Canceled)

4. (Currently Amended) A method as recited in claim 1, wherein a ~~the~~ character sequence in the word has a first number of multiple characters and a ~~the~~ character sequence in the string has a second number of multiple characters that is different from the first number of multiple characters.

5. (Original) A method as recited in claim 1 and further comprising determining how likely the word is to have been generated.

6. (Currently Amended) A method as recited in claim 1 and further comprising conditioning an ~~the~~ edit operation that changes the string into the word on at least one of the probabilities. ~~a position that the edit occurs at within the word.~~

7. (Original) A method as recited in claim 1 and further comprising identifying the string as potentially incorrect.

8. (Original) A method as recited in claim 1 and further comprising correcting the string to the word.

9. (Original) A computer readable medium having computer-executable instructions that, when executed on a processor, perform the method as recited in claim 1.

10. (Currently amended) A method comprising:

receiving an entered string *s*; and

determining a probability $P(s|w)$ expressing how likely a word *w* was to have been incorrectly entered as the string *s* based on partitioning the word *w* and the string *s* and computing probabilities for various partitionings, wherein a probability for a partitioning represents the probability that one or more edit operations ~~that~~ convert first arbitrary-length character sequences $\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_n$ in the word *w* to corresponding second arbitrary-length character sequences $\beta_1, \beta_2, \beta_3, \ldots, \beta_n$ in the string *s*, wherein:

$$P(s|w) = P(\beta_1|\alpha_1) * P(\beta_2|\alpha_2) * P(\beta_3|\alpha_3) *\ldots *P(\beta_n|\alpha_n)$$

11. (Original) A method as recited in claim 10, wherein lengths of corresponding first and second character sequences are different.

12. (Original) A method as recited in claim 10 and further comprising determining how likely the word *w* is to have been generated.

13. (Original) A method as recited in claim 10 and further comprising conditioning the edit operations on positions that the edits occur at within the word.

14. (Original) A method as recited in claim 10 and further comprising correcting the string *s* to the word *w*.

15. (Original) A method as recited in claim 10 and further comprising identifying the string *s* as potentially incorrect.

16. (Original) A computer readable medium having computer-executable instructions that, when executed on a processor, perform the method as recited in claim 10.

17. (Original) A method comprising:

receiving an entered string *s*; and

determining a probability $P(s|w)$ expressing how likely a word *w* was to have been incorrectly entered as the string *s*, by partitioning the word w and the string s and computing probabilities for various partitionings, as follows:

$$P(s \mid w) = \sum_{R \in Part(w)} P(R \mid w) \sum_{\substack{T \in Part(s) \\ |T|=|R|}} \prod_{i=1}^{|R|} P(T_i \mid R_i)$$

where Part(w) is a set of possible ways of partitioning the word w, Part(s) is a set of possible ways of partitioning the string s, R is a particular partition of the word w, and T is a particular partition of the string s.

18. (Original) A method as recited in claim 17 and further comprising selecting the partition that returns a highest probability.

19. (Original) A method as recited in claim 17 and further comprising determining how likely the word w is to have been generated.

20. (Original) A method as recited in claim 17 and further comprising correcting the string s to the word w.

21. (Original) A method as recited in claim 17 and further comprising identifying the string s as potentially incorrect.

22. (Original) A computer readable medium having computer-executable instructions that, when executed on a processor, perform the method as recited in claim 17.

23. (Original) A method comprising:

receiving an entered string *s*; and

determining a probability P(*s*|*w*) expressing how likely a word *w* was to have been incorrectly entered as the string *s*, by partitioning the word w and the string s and computing probabilities for various partitionings, as follows:

$$P(s|w) = \max_{R \in Part(w),\, T \in Part(s)} P(R|w) * \prod_{i=1}^{|R|} P(T_i | R_i)$$

where Part(*w*) is a set of possible ways of partitioning the word *w*, Part(*s*) is a set of possible ways of partitioning the string *s*, R is a particular partition of the word *w*, and T is a particular partition of the string *s*.

24. (Original) A method as recited in claim 23 and further comprising omitting the term P(R|w) from the computation of P(*s*|*w*).

25. (Original) A method as recited in claim 23 and further comprising setting terms $P(T_i|R_i) = 1$ whenever $T_i = R_i$.

26. (Original) A method as recited in claim 23 and further comprising determining how likely the word *w* is to have been generated.

27. (Original) A method as recited in claim 23 and further comprising correcting the string *s* to the word *w*.

28. (Original) A method as recited in claim 23 and further comprising identifying the string *s* as potentially incorrect.

29. (Original) A computer readable medium having computer-executable instructions that, when executed on a processor, perform the method as recited in claim 23.

30. (Original) A method comprising:

receiving an entered string *s*; and

determining a probability P(*s*|*w*) expressing how likely a word *w* was to have been incorrectly entered as the string *s,* by partitioning the word w and the string s and finding a partition R of the word w and a partition T of the string s such that $\prod_{i=1}^{|R|} P(T_i \mid R_i)$ is maximized.

31. (Original) A method as recited in claim 30 and further comprising determining how likely the word *w* is to have been generated.

32. (Original) A method as recited in claim 30 and further comprising correcting the string *s* to the word *w*.

33. (Original) A method as recited in claim 30 and further comprising identifying the string *s* as potentially incorrect.

34. (Original) A computer readable medium having computer-executable instructions that, when executed on a processor, perform the method as recited in claim 30.

35. (Original) A method for training an error model used in a spell checker, comprising:

determining, given a <wrong, right> training pair and multiple single character edits that convert characters in one of the right or wrong strings to characters in the other of the right or wrong strings at differing costs, an alignment of the wrong string and the right string that results is a least cost to convert the characters;

collapsing any contiguous non-match edits into one or more common error regions, each error region containing one or more characters that can be converted to one or more other characters using a substitution edit; and

computing a probability for each substitution edit.

36. (Original) A method as recited in claim 35, wherein the assigning comprises assessing a cost of 0 to all match edits and a cost of 1 to all non-match edits.

37. (Original) A method as recited in claim 35, wherein the single character edits comprises insertion, deletion, and substitution.

38. (Original) A method as recited in claim 35, further comprising collecting multiple <wrong, right> training pairs from online resources.

39. (Original) A method as recited in claim 35, further comprising expanding each of the error regions to capture at least one character on at least one side of the error region.

40. (Previously Presented) A program embodied on a computer readable medium, which when executed, directs a computer to perform the following:

receive an entered string; and

determine how likely an expected string was to have been entered as the entered string based on at least one edit operation that converts one of multiple character sequences of arbitrary length in the expected string to one of multiple character sequences of arbitrary length in the entered string.

41. (Previously Presented) A program as recited in claim 40, wherein the one of multiple character sequences of the expected string has a first length and the one of multiple character sequences of the entered string has a second length that is different than the first length.

42. (Previously Presented) A program as recited in claim 40, wherein the one of multiple character sequences of the expected string has multiple characters and the one of multiple character sequences of the entered string has multiple characters.

43. (Previously Presented) A program as recited in claim 40, wherein the one of multiple character sequences of the expected string has a first number of multiple characters and the one of multiple character sequences of the entered string has a second number of multiple characters that is different from the first number of multiple characters.

44. (Original) A program as recited in claim 40, further comprising computer-executable instructions that directs a computer to determine how likely the expected string is to have been generated.

45. (Previously Presented) A program as recited in claim 40, further comprising computer-executable instructions that directs a computer to perform, depending upon how likely an expected string was to be incorrectly entered as the entered string, one of leaving the entered string unchanged, autocorrecting the entered string into the expected string, or offering a list of possible corrections.

46. (Original) A spell checker program, embodied on a computer-readable medium, comprising the program of claim 40.

47. (Original) A language conversion program, embodied on a computer-readable medium, comprising the program of claim 40.

48. (Original) A word processing program, embodied on a computer-readable medium, comprising the program of claim 40.

49. (Currently Amended) A program embodied on a computer readable medium, which when executed, directs a computer to perform the following:

(a) receive an entered string *s*;

(b) for multiple words *w* in a dictionary, determine:

how likely a word *w* in a dictionary is to have been generated, P(*w*|*context*); and

how likely the word *w* was to have been entered as the string *s*, P(*s*|*w*), based on <u>partitioning the word *w* and the string *s* and computing probabilities for various partitionings to determine a highest likelihood of</u> at least one edit operation that converts one of multiple character sequence of arbitrary length in the word to one of multiple character sequences of arbitrary length in the string; and

(c) maximize P(*s*|*w*)*P(*w*|*context*) to identify which of the words is most likely the word intended when the string s was entered.

50. (Previously Presented) A program as recited in claim 49, wherein the determination (b) is performed for all words in the dictionary.

51. (Previously Presented) A program as recited in claim 49, further comprising computer-executable instructions that directs a computer to perform one of leaving the string unchanged, autocorrect the string into the word, or offer a list of possible corrections.

52. (Original) A spell checker program, embodied on a computer-readable medium, comprising the program of claim 49.

53. (Original) A language conversion program, embodied on a computer-readable medium, comprising the program of claim 49.

54. (Currently Amended) A spell checker comprising:

a source model component to determine how likely a word *w* in a dictionary is to have been generated; and

an error model component to determine how likely the word *w* was to have been incorrectly entered as the string *s* based on arbitrary length string-to-string transformations<u>, wherein the error model partitions the word *w* and the string *s* and computes probabilities for various partitionings</u>.

55. (Original) A spell checker as recited in claim 54, wherein the string-to-string transformations involve conversion of a first character sequence of a first length into a second character sequence of a second length that is different than the first length.

56. (Original) A spell checker as recited in claim 54, wherein the string-to-string transformations involve conversion of a first character sequence with multiple characters into a second character sequence with multiple characters.

57. (Original) A spell checker as recited in claim 54, wherein the string-to-string transformations involveconversion of a first character sequence having a first number of multiple characters into a second character sequence having a second number of multiple characters that is different from the first number of multiple characters.